On The Optimality of Adaptive Allocation of Excess Available Bandwidth by a Guaranteed Rate (GR) Diff-Serve QoS Scheduler in an Ad-hoc Mobile Network

Avik Chakraborty
e-mail: avik.chakraborty@gmail.com
Mentor Graphics (NOIDA) Pvt. Ltd.
A-4, Sector-10, Noida, UP, India

Abstract:

The Guaranteed Rate (GR) scheduling schemes such as Weighted Round Robin (WRR) provide hard end-to-end delay bound for leaky-bucket constrained traffic and ensure fair allocation of bandwidth among all backlogged sessions for any traffic. In ad-hoc mobile networks, the bandwidth requirement is high as the traffic is bursty in nature and each node is responsible for managing overall Quality of Service (QoS) requirements unlike Hierarchical Mobile Network. We propose to adaptively distribute residual bandwidth among the backlogged flows by taking into account the correlation property of traffic stream. As ad-hoc mobile network traffic is self-similar in nature and aggregating such flows at the output queues of a Diff-Serve scheduler strengthens the self-similarity (i.e. Long-range Dependence (LRD)), it is one-step ahead forecastable by ARIMA (Auto-regressive Integrated Moving Average) model. ARIMA model is suitable for prediction of traffic with high LRD/SRD ratio. But, finite buffer queues limit the effect of auto-correlations of self-similar traffic. It implies that we need to capture short-range dependence (SRD) of the arriving traffic as well. On the other hand, during congestion we need to ensure that the scheduler is also adaptive to the packet-loss as we intend to avoid the computational and network overhead incurred by conventional control-messaging techniques (for ex. RED (Random Early Detection), BLUE etc.) that have been used so far to overcome packet-loss. Hence, we propose to distribute the residual bandwidth based on share-weight factors of output queues. Share-weight factors are calculated from a convex combination of ARIMA predicted traffic, Least Slack (i.e. Due-date – Worst case delay experienced by the last packet) and the packet-loss at the previous time-window. Our simulation results show that our scheduling algorithm outperforms other algorithms and achieves optimal scheduling to ensure better QoS. For traffic with small-amplitude random noise component, once it reaches optimal operating point, it ensures almost totally loss-less scheduling of packets.

1. Introduction:

Quality of Service (i.e. QoS) aims to effectively arrange the network resources so that different network services can meet their service requirements as per the service level agreements (i.e. SLAs). But, Bandwidth is a critical and limited resource compared to exponential increase in the wireless data transmission and advent of multimedia applications that require much more bandwidth than normal application. It becomes even more difficult in an ad-hoc mobile network where user is continuously moving from one cell to another. Obviously, best-effort mechanism is not a solution. The different QoS requirements that are directly experienced by a mobile user under an ad-hoc mobile network can be broadly classified into two categories: 1. Connection-level QoS associated with each flow and 2. Packet-level QoS associated with each service class. The first category includes IntServe QoS for ex. Seamless hand-off, Probability of Connection Blocking, Graceful Degradation (degradation frequency) etc. and the second category takes into account DiffServe QoS for ex. End-to-end Delay bound, Delay variance (jitter), packet-loss rate etc. As each node is responsible for overall QoS, this necessarily gives rise to the notion of a two-stage scheduler where the first stage allocates bandwidth to each flow based on its IntServe requirements and the second stage aggregates flows belonging to the same service class and allocates the excess available bandwidth to each output queue based on some criteria. As number of service classes is finite unlike number of flows, DiffServe is certainly scalable. Based on DS field in the IP header IP flows are classified into different aggregates and services are provided to aggregates based on small set of forwarding behaviors called PHBs (Per-hop behavior). Some researchers have already proposed some algorithms as possible solution to the problem of adaptive QoS scheduling. To mention a few Weighted Fair Queuing (WFQ) [1], Virtual Clock (VC) [2], Rate-controlled Earliest Deadline First (RC-EDF) [3], Class-based Queuing (CBQ) (here EF packets are given priority upto the configured Expedited Forwarding rate) [4], Differential Multi-layer Gated Frame Queuing (DMGFQ) [5] but none of them actually addressed the problem of excess available bandwidth allocation in a two-stage scheduler. To deal with the traffic distortion and dynamics of flow aggregation Wang etal have [6] proposed an adaptive-weighted packet-scheduling scheme, which can be applied to weighted round robin or fair queuing. They try

to adjust the sizes of output queues based on prediction made from EWMA (Exponentially Weighted Moving Average) model unlike ours where we propose to dynamically allocate residual bandwidth to fixed-size output queues.

2. Characteristics of Ad-hoc Mobile Network Traffic:

Ad-hoc mobile network traffic is highly self-similar in nature in nature and aggregating such traffic streams strengthens the self-similarity. The implication is that if each node locally predicts future traffic one-step ahead by using real-time and historical information, all nodes co-operate to achieve overall QoS in the whole network. Leland etal [7] has shown that Ethernet LAN traffic is statistically self-similar in nature and they have proven that ARIMA model is the most suitable for predicting such self-similar data streams. Corradi etal's study [8] suggests that fractional ARIMA model is able to capture both LRD (i.e. Long-range Dependence) and SRD (i.e. Short-range Dependence) features of the traffic. Indeed f-ARIMA model is able to predict a self-similar traffic with high LRD/SRD component ratio, but its inadequacy is reflected while predicting traffic with a complex SRD component (for ex. Random noise instead of Gaussian White Noise (WN)). This is in contrast with the traditional traffic models all of which exhibit SRD property. Different LRD models have been considered in the framework of traffic engineering involving exactly (ex. FGN i.e. fractional Gaussian Noise) as well as asymptotically self-similar processes (ex. Chaotic map, Heavy-tailed ON-OFF models, f-ARIMA). Among them f-ARIMA (p,d,q) is the most general and flexible solution since it permits to fit well separately the short and long term behavior of the auto-covariance function of the f-ARIMA process. F-ARIMA prediction model has been used to build a CAC scheme [9], and a dynamic bandwidth allocation scheme [10]. In this dynamic allocation scheme, the predicted traffic values are used directly regardless of the buffer length, and that may result in buffer overflow. Takahashi etal [11] have however proven ARIMA model's superiority over f-ARIMA model. To compare the SRD and LRD models, they used the AIC (Akaike's Information Criterion) (BIC (i.e. Bayesian Information Criterion) can be used instead). The conclusion above derives the suggestion that it will be better to use the ARIMA than f-ARIMA model in rough traffic simulations, which is opposite to the ideas believed widely so far. A Linear Dynamic system with self-similarity (high LRD/SRD component ratio) can be modeled by ARIMA model. If the effect (i.e. shock) of X_{t-j} on Y_t is e_j, then,

$Y_t = e_0X_t + e_1X_{t-1} + e_2X_{t-2} + \ldots\ldots + e_jX_{t-j} + \ldots\ldots$

ARMA (p.q) model combines *AR (p)* and *MA (q)* models where *p = order of auto-regression* and *q = order of moving average*.

$Y_t - f_1Y_{t-1} - f_2Y_{t-2} - \ldots\ldots - f_pY_{t-p} = C + a_t - g_1a_{t-1} - g_2a_{t-2} - \ldots\ldots - g_qa_{t-q}$

If the level of successive differencing is *d*, then the model is called *ARIMA (p,d,q).*

B is the back-ward shift operator such that $BX_t = X_{t-1}$

The differencing operator is $\nabla = 1 - B$ such that $\nabla X_t = X_t - X_{t-1}$

$\nabla^d = (1 - B)^d = \sum_{k=0}^{\infty} {}^dC_k\ (-B)^k = \sum_{k=0}^{d} {}^dC_k\ (-B)^k$ where ${}^dC_k = d! / k!\ (d-k)!$

$f(B) = 1 - f_1B - f_2B^2 - f_3B^3 - \ldots\ldots - f_pB^p$

$g(B) = 1 - g_1B - g_2B^2 - g_3B^3 - \ldots\ldots - g_qB^q$

If X_t follows *ARIMA (p,d,q)* process then,

$f(B)\nabla^dX_t = g(B)a_t$ where a_t is the white noise i.e. a_t follows Gaussian distribution with mean $\mu = 0$ and variance σ^2.

3.The Scheduling Algorithm:

If there are *m* service types each with p_i service classes and all combinations are possible, then number of output queues is equal to $p_1 \times p_2 \times p_3 \ldots\ldots\ldots p_m$.

Let total available bandwidth (i.e. Link Capacity) be equal to *G*.

Let no. of flows of type *i* is N_i, then the per-flow processor assigns guaranteed bandwidth for the j-th flow $g^{si}{}_j$

$\Rightarrow$ Guaranteed Bandwidth for the output queue associated with service class i is $g_i = \Sigma_j g^{si}{}_j$

Excess available bandwidth beyond guaranteed load $EABw = G - (\Sigma_i \Sigma_j g^{si}{}_j)$

If normalized share-weight factor associated with output queue of service class *i* is s_i, then actual bandwidth allocated output queue *i* is $\Sigma g^{si}{}_j + s_i \times EABw = g_i + s_i \times EABw$

Scheduling Time Interval:-

Minimum allocated bandwidth to any queue = $min_i\ (g_i)$.

and let scheduling time interval be *T*.

We assume that at least one packet is transmitted each time a queue gets its turn.

$\Rightarrow$ if τ is the transmission time for a packet, then

$T \times min_i\ (g_i) / G \geq \tau$

$\Rightarrow$ If *T* is set to be as less as possible then, $T = (G \times \tau) / min_i\ (g_i)$;

Share-weight factors that are calculated by the share-weight factor estimator over the time interval nT to $nT + T$ is used at the beginning of $nT + T$.

Handling Bursty Traffic:-
As output queues are of finite sizes, bursty traffic leads to overflow of packets.
$\Rightarrow$ This increases the packet-loss rate.
Hence, it also decreases the reliability of QoS.
This can actually be overcome by incorporating adaptability to internal queue configurations i.e. Least Slack (Due-date – Worst-case Delay experienced by the last packet).
Its advantage is two-fold:
1.Reduces delay-variance (i.e. jitter)
2.Takes into account the short-range dependence (for ex. in the form of random noise) of the arriving traffic stream. ARIMA model captures Long-range Dependence (LRD) of the self-similar traffic with high LRD/SRD ratio. Hence, if share-weight factors are calculated from a convex combination of ARIMA predicted flow and Least Slack, then both LRD and SRD dependencies can be taken care of.
Handling Packet-losses in times of Congestion:-
For relatively small buffer sizes, in case of a traffic stream with high SRD in the form of high amplitude noise, the LSs do not necessarily reflect the priority values that are to be assigned to the queues. Instead, the queues that are under heavy traffic load much larger than the bandwidth allocated to it, suffer from very heavy rate of packet-loss. There are call admission control (CAC) and traffic policing techniques for ex. Random Early Detection (*RED*), *BLUE* etc. to overcome packet-losses during congestion. These are all control messaging techniques that cost messaging delay and consequent initial packet losses. Instead we aim to handle congestion in the scheduling scheme itself. This necessarily implies that share-weight factors should also be calculated from packet-loss priorities. We therefore conclude that share-weight factors should be calculated from a convex combination of ARIMA predicted traffic flow, Least Slack and packet-loss rate.
Share-weight factor calculation:-
Using ARIMA model the traffic flow for the time interval $(nT + T)$ is calculated by using the past history as follows,
$Y_t = f_1 Y_{t-1} + f_2 Y_{t-2} + \ldots\ldots + f_p Y_{t-p} + C + a_t - g_1 a_{t-1} - g_2 a_{t-2} - \ldots\ldots - g_q a_{t-q}$
For a given ad-hoc mobile network traffic trace, the parameters $f_1, f_2 f_3, \ldots\ldots, f_p, g_1, g_2, g_3, \ldots\ldots\ldots, g_q$ are calculated by ARIMA model estimation techniques that are discussed below.
If $X_j^{(nT+T)}$ denotes the predicted traffic flow for the j-th output queue, then ARIMA priority component of queue j is :
$p_j^{ARIMA} = X_j^{(nT+T)} / (\Sigma_j X_j^{(nT+T)})$
Let the minimum allocated bandwidth (time slice in an RR scheduler) to output queue j be at least $g_j \times T/G$
Let number of packets in the j-th queue be $(n_j + 1)$.
Then, worst case delay experienced by the last packet is :
$((n_j + 1)/((g_j \times T)/(G \times \tau))) \times T + T = (n_j + 1) \times G \times \tau / g_j + T$
$\Rightarrow$ Least Slack for the j-th output queue i.e. $LS_j = (Due\text{-}date_j - (n_j + 1) \times G \times \tau / g_j - T)$
Then LS priority value associated with output queue j is calculated as:
$p_j^{LS} = (\Sigma_j LS_j^{(nT+T)}) / LS_j^{(nT+T)}$
Now consider the packet-loss priority calculation. If packet loss of queue j at the end of the interval nT is : $packet\text{-}loss_j$
Then, $p_j^{packet\text{-}loss} = Packet\text{-}loss_j^{(nT)} / (\Sigma_j Packet\text{-}loss_j^{(nT)})$
This is used at the beginning of interval $(nT + T)$.
Share-weight factor associated with output queue j i.e. s_j is calculated from a convex combination of p_j^{ARIMA}, p_j^{LS} and $p_j^{packet\text{-}loss}$ as follows:
$s_j = \alpha_{ARIMA} \times p_j^{ARIMA} + \alpha_{LS} \times p_j^{LS} + \alpha_{packet\text{-}loss} \times p_j^{packet\text{-}loss}$ where $0 \le \alpha_{ARIMA}, \alpha_{LS}, \alpha_{packet\text{-}loss} \le 1$
and $\alpha_{ARIMA} + \alpha_{LS} + \alpha_{packet\text{-}loss} = 1$ Finally share-weight factors are normalized as $s_j \Leftarrow s_j / (\Sigma_j s_j)$

4. Simulation Results:

For simulation we have assumed an output-queued scheduler with output queues and we have assumed rather a more general scenario where each of the queues is guaranteed a certain amount of bandwidth and we show that our algorithm outperforms other algorithms in terms of 2 QoS parameters: 1. Rate of overall packet-loss less intervals and 2. Overall weighted fairness as defined subsequently. We have simulated the performance over time interval of 200 time-units.
We have assumed that the aggregated traffic at each of three queues follows the following traffic models y(t), 2y(t) and 3y(t), (Differenced time series: y(t) = (1-L) {ARIMA test data} Model: y(t) = b(1)x(1) + u(t),
where u(t) is an ARMA process and x(1) = 1
Model for u(t): [1 - a(1,1)L]u(t) = [1 - a(2,1)L][1 - c(2,1)L^4]e(t), with e(t) white noise, and L the lag operator.)

We have run the WFQ scheme with fixed weights with ratio 1:2:3 as share-weight factors without any adaptation, though it is intrinsically a completely predictive scheme as it correctly estimates the future traffic as the ratio of arriving

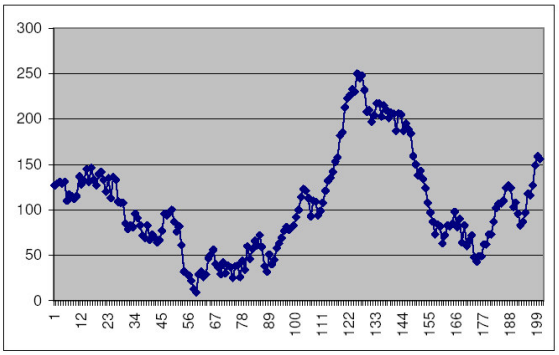
traffic streams is 1:2:3. In our proposed scheme we have predicted the one-step ahead future traffic using the ARIMA model, and calculated the share-weight factors from a convex combination of ARIMA predicted traffic flow, Least Slack (LS) of each queue and packet-losses at each queue.
The total available bandwidth (i.e. link capacity) G = 400
Guaranteed Bandwidths at each queue $g_1 = 50$, $g_2 = 100$, $g_3 = 150$
Excess Available Bandwidth EABw = 400 – (50 + 100 + 150) = 100
In WRR, the share-weight factors are fixed: $s_1 = 0.16$, $s_2 = 0.33$, $s_3 = 0.51$
For simulation, we have generated the traffic at each queue as follows:
$y(t) + A_1 \times r(t)$, $2y(t) + A_2 \times r(t)$ and $3y(t) + A_3 \times r(t)$ where we have varied the random noise amplitudes A_1, A_2 and A_3 (r(t) generates a floating point between 0.0 and 1.0 totally randomly i.e. actually pseudo-random generator rand() of GNU C Library). In our scheme we have varied the convex combination parameters α_{ARIMA} ,α_{LS} and $\alpha_{PACKET\text{-}LOSS}$ to find out optimal combination .
Our results show the following results:
1. When the noise amplitudes are small and buffer sizes are greater than the sum of average traffic and noise amplitude, then the factor α_{ARIMA} should be the most dominant.
2. When the noise amplitudes are large enough and the buffer sizes are greater than the sum as before, then both α_{ARIMA} and α_{LS} should be chosen as the dominant factors.
3. When the noise amplitudes are significantly large and the buffer sizes are smaller than the sum of average traffic and noise amplitudes, then the factor $\alpha_{PACKET\text{-}LOSS}$ should be the most dominant.
We have also calculated the five QoS parameters by fixing the convex combination parameter values as follows:
$\alpha_{ARIMA} = 1.0$, $\alpha_{LS} = 0.0$, $\alpha_{PACKET\text{-}LOSS} = 0.0$ (WFQ; completely predictive)
$\alpha_{ARIMA} = 0.0$,$\alpha_{LS} = 1.0$, $\alpha_{PACKET\text{-}LOSS} = 0.0$ (adaptive only to due-date slack)
$\alpha_{ARIMA} = 0.0$,$\alpha_{LS} = 0.0$, $\alpha_{PACKET\text{-}LOSS} = 1.0$ (adaptive only to packet-loss rate)
We have demonstrated the results for two kind of traffic, one with small noise amplitudes and another one with large noise amplitudes. We have calculated "fairness" value of each queue as follows:
Fairness of queue i = (Σ_t (actually transmitted bandwidth / guaranteed bandwidth)) / Σ t
We have calculated overall weighted fairness using the following formula:
Overall Fairness = $(g_1 / (g_1 + g_2 + g_3))$ $Fairness_1$ + $(g_2 / (g_1 + g_2 + g_3))$ $Fairness_2$ +
$(g_3 / (g_1 + g_2 + g_3))$ $Fairness_3$;
Two kinds of scenarios that have been assumed are as follows: Queue SIZES: 60 120 170; TOTAL LINK B/W: 400
GUARANTEED LOADS: 50 100 150; DUE-DATES: 500.000000 550.000000 600.00000
NOISE BOUNDS: in one case 10 20 30; and in another case 40 70 100
Fig 1 and Fig 2 show traffic trace at the largest queue with small and large amplitude random noise component respectively. Fig 3 and 4 show trace of packet-loss in the largest queue for above 2 cases. Fig 5 shows the comparison of overall lossless intervals among 1.optimal 2.predictive 3.adaptive 4. packet-loss adaptive in case of small noise. Fig 6 shows the comparison of overall weighted fairness for the same case. Fig 7 and 8 demonstrate the comparisons in case of large amplitude random noise component.

References:

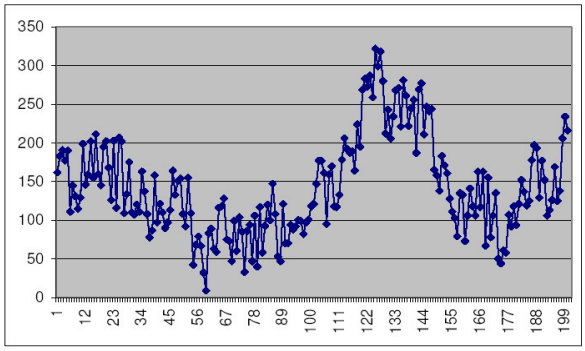
[1] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm", Proceedings of ACM SIGCOMM'89, September,1989.
[2] L. Zhang, "Virtual Clock: A New Traffic Control Algorithm for Packet Switching Networks", Proceedings of ACM SIGCOMM'90, September,1990.
[3] H. Zhang and D. Ferrari, "Rate-controlled Static-priority Queueing", Proceedings of IEEE INFOCOM'93, April, 1993.
[4] S. Floyd and V. Jacobson, "Link-sharing and Resource Management Models for Packet Networks" IEEE/ACM Transactions on Networking, Vol. 3, No. 4, August 1995.
[5] H. B. Chiou, F. M. Tsou, Z. Tsai, "DMGFQ: A Novel Traffic Scheduler with Differentiated QoS Guarantee for Internet Multimedia Services", IEEE ICC'2001, June 11-15, 2001.
[6] H. Wang, C. Shen, K. Shin, "Adaptive-weight Packet Scheduler for Supporting Premium Service,"IEEE International Conference on Communications'2001 JUNE 2001, HELSINKI, FINLAND
[7] On the Self-similar Nature of Ethernet Traffic (Extended Version) W. E. Leland, M. S. Taqqu, W. Willinger, D. V. Wilson
[8] M. Corradi, R. G. Garroppo, S. Giordano and M. Pagano, "Analysis of f-ARIMA processes in the modeling of broadband traffic", ICC'01, vol. 3, pp. 964-968, 2001.

[9] Y. Shu, Z. Jin, J. Wang and O. W. W. Yang, “Prediction-based admission control using FARIMA models”, ICC’00, vol. 2, pp. 1325-1329, 2000.
[10] Z. Jin, Y. Shu, J. liu and O. W. W. Yang, “Prediction-based network bandwidth control”, Canadian Conference on Elec. And Comp. Eng., vol. 2, pp. 675-679, 2000.
[11] ARIMA model's superiority over f-ARIMA model, *Takahashi, Y.; Aida, H.; Saito, T.;* Communication Technology Proceedings, 2000. WCC - ICCT 2000. International Conference on , Volume: 1 , 21-25 Aug. 2000 Pages:66 - 69 vol.1

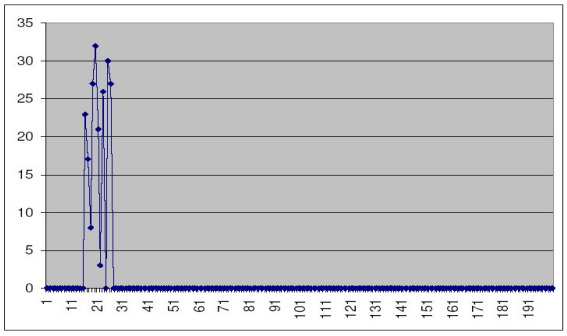
(1)



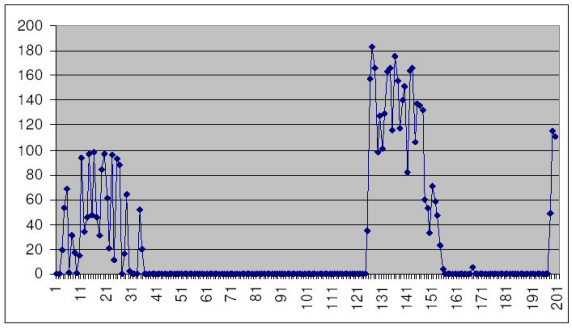
(2)



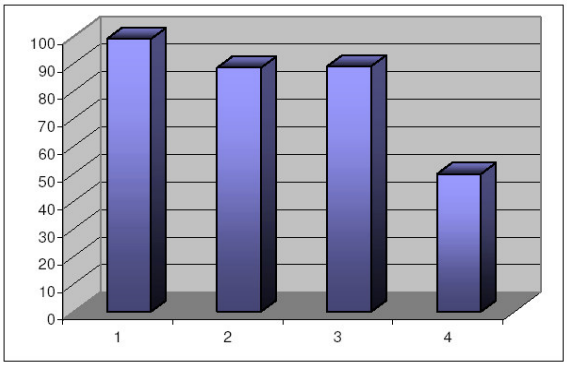
(3)



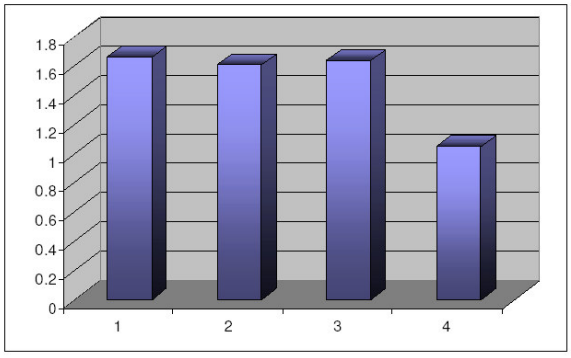
(4)



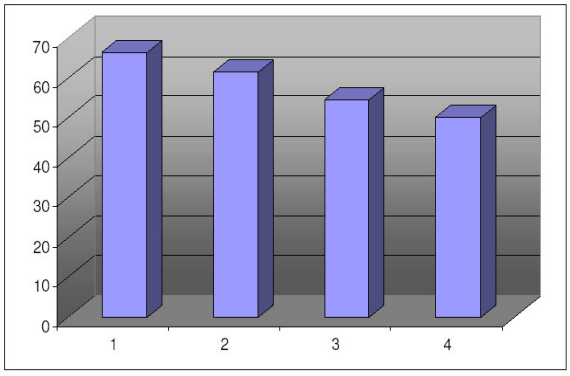
(5)



(6)



(7)



(8)